総合報告

整数/組合せ計画法の現状（その5）
ナップザック問題および集合被覆(分割)問題

整数計画法研究部会　鈴木久敏・岩村覚三

はじめに

今回は，ナップザック問題(knapsack problem)，集合被覆問題(set covering problem)，集合分割問題(set partitioning problem)の三つをサーベイする．これらの問題は，整数計画問題の中でも比較的単純な構造をしているため，その構造の特殊性を利用した，あるいはデータ構造に工夫をこらした強力なアルゴリズムが開発されている．

第1部　ナップザック問題

1.1　ナップザック問題とは

ナップザック問題は1957年に Dantzig[6] によって提唱され，「ハイキングに出かける際に，重量制約(サイズ)が b のナップザックに詰め込みたい n 種類の品物があり，各品物 j($j=1,\cdots,n$) の重量 a_j と価値 c_j が既知のとき，総価値を最大とする品物の組合せは何か」という問題である．これが「ナップザック問題」という名の由来であり，数理的には，

0-1ナップザック問題 P_0：

$$\max \quad z_0=\sum_{j=1}^{n} c_j x_j \tag{1.1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} a_j x_j \leqq b \tag{1.2}$$

$$x_j=0 \text{ or } 1,\ j=1,\cdots,n \tag{1.3}$$

と単一制約の 0-1 整数計画問題に定式化される．$x_j=1$ は品物を詰め込むこと，$x_j=0$ は詰め込まないことを意味する．一般性を失わずに，係数 a_j，c_j，b は正と仮定できる．さらに，計算の便宜等の理由から，a_j および b は整数と仮定する場合も多々ある．しかしながら，係数が整数でも「問題の複雑さ(難しさ)」は本質的に変らず，問題は依然としてNP完全である(詳細は次回)．

0-1変数の条件(1.3)の代りに，

$$x_j \geqq 0,\ x_j \text{ は整数},\ j=1,\cdots,n \tag{1.4}$$

で置き換えた問題を「ナップザック問題」とよぶ場合も多く，本報告ではこの問題を整数値ナップザック問題 P (integer knapsack problem)とよび，先の0-1ナップザック問題と区別する．a_j，$b>0$ より，変数 x_j は有界($0\leqq x_j\leqq[b/a_j]$，[ ] はガウス記号) となり，x_j を有限個の 0-1 変数で置き換えれば，問題 P を等価な 0-1 ナップザック問題 P_0 に書き換えることができる．

容積，コストなどその他の資源に関する制約があり，複数の制約式をもつ場合を，多次元ナップザック問題(multidimensional knapsack problem)と言う．

本報告においては，とくに断わらないかぎり

ナップザック構造

a）目的関数，制約式が線形

b）単一制約

c）正係数(または，正の整数係数)

の三つの特殊構造をもつ問題 P_0 および P を対象とする．

Salkin & Dekluyver[33] と Salkin[32] では，1973年以前のナップザック問題の研究が，比較的詳細にサーベイされている．

1.2　ナップザック問題の重要性

整数計画法(以下 ILP と略称)の分野で，ナップザック問題がとくに詳しく研究されてきた理由として，つぎの四つが考えられる．

1) ILPの中でもっとも単純なモデル構造をもち，問題解決の理論的見通しが立てやすいにもかかわらず，ILPの本質的難しさを内在させており，研究対象としての価値が高い．問題構造の単純性に関して，Lauriere[25] は，(i)目的関数の上界値の算出が容易，(ii)下界値を与える実行可能解の生成が容易の2点を指摘している．この性質は，多変数の大規模問題を分

枝限定法で解くとき，たいへん有効となる．これは，前述のナップザック構造に強く依存した特徴であり，後で詳しく述べることにする．

2) 任意の ILP(注1)は，複数の制約式を実行可能解集合 X_I が等しい単一制約式に集約(aggregation)することにより，等価なナップザック問題に変換できる．この集約に要する計算量が微かならば，与えられた ILP の代りに，構造が単純で強力なアルゴリズムが開発されている等価なナップザック問題を解けば良いわけである([4][12][19]など)．

3) ある種の数理計画問題を解くとき，その部分問題としてナップザック問題を解く必要が生じる．例としては，Gilmore & Gomory[10]が材料切断問題を列生成で解いた場合，Pierce[30]の集合分割問題の場合，Kianfar[20]の強力な切除平面の生成の場合などを挙げることができる．

4) 資本予算編成問題[36]，信頼性冗長問題[35]，図書館の購入雑誌選択[23]，試験問題の構成と採点[8]など現実面への応用例が豊富である．

つぎに，1)で述べた目的関数の上界値および下界値の算出法を説明しよう．対象として，0-1 ナップザック問題 P_0 を取り上げる．いま，n 個の変数を，

$$\rho_1 \geqq \rho_2 \geqq \cdots \geqq \rho_n, \quad \text{ただし } \rho_j = c_j/a_j \tag{2.1}$$

が満たされるように並べ替える．これは単一制約であるから可能となるのであって，この並べ替えには $O(n \log n)$ の計算量を必要とする．

元問題 P_0 に対して，0-1 変数条件(1.3)を外して，

$$0 \leqq x_j \leqq 1, \quad j=1,\cdots,n \tag{2.2}$$

を満たす連続変数で置き換えた連続緩和問題 $\bar{P}_0$ を考える．$\bar{P}_0$ は形式的には LP 問題であるが，以下のように $O(n)$ の計算量で解ける．整数 $p(1 \leqq p \leqq n)$ が，

$$\sum_{j=1}^{p-1} a_j \leqq b < \sum_{j=1}^{p} a_j \tag{2.3}$$

を満たすとき(注2)，$\bar{P}_0$ の最適解 $\bar{x}_j(j=1,\cdots,n)$ は，$\bar{x}_j=1(j<p)$，$\bar{x}_p=(b-\sum_{j=1}^{p-1} a_j)/a_p$，$\bar{x}_j=0(j>p)$ で与えられる．$\bar{x}_p$ をピボット変数とよび，(2.3)の条件より $0 \leqq \bar{x}_p < 1$ である．

緩和問題 $\bar{P}_0$ の最適値は，元問題 P_0 の最適値 z_0^* に対して一つの上界値 z_0^u となるから，

$$z_0^u = \sum_{j=1}^{p-1} c_j + c_p \bar{x}_p \tag{2.4}$$

である．一方，ピボット変数 $\bar{x}_p$ を0に切り下げた解，すなわち $\bar{x}_j=1(j<p)$ かつ $\bar{x}_j=0(j \geqq p)$ は，元問題 P_0 の一つの実行可能解となる．対応する目的関数の値が z_0^* の下界値 z_0^l となり，$z_0^l = \sum_{j=1}^{p-1} c_j = z_0^u - c_p \bar{x}_p$ である．$\bar{x}_p=0$ のときは，$\bar{P}_0$ の最適解が P_0 の最適解となる．$\bar{x}_p>0$ のときは，ナップザックのサイズにまだ $b'=a_p \bar{x}_p$ だけ余裕が残っており，これを利用して $j>p$ なる品物を詰めることができれば，さらに良い下界値が得られるはずである．そのような下界値 z_0^l は，

貪欲解法(greedy algorithm)：

$$g_j(y) = c_j[y/a_j] + g_{j+1}(y \bmod a_j), \quad j=1,\cdots,n$$
$$\text{ただし，} g_{n+1}(y)=0 \tag{2.5}$$

なる手続きで求めた $g_1(b)$ の値に等しい．よって，

$$z_0^l = g_1(b) \tag{2.6}$$

である．貪欲解法は $O(n)$ の計算量で $g_1(b)$ を得る．

1.3 アルゴリズム

ダイナミック・プログラミング(DP)

DPを用いてナップザック問題を解くときは，

$$F_k(y) = \max_{x_1,\cdots,x_k} \left\{ \sum_{j=1}^{k} c_j x_j \,\middle|\, \sum_{j=1}^{k} a_j x_j \leqq y,\ x_j=0,\cdots,u_j \right\} \tag{3.1}$$

なる部分問題を考えると便利である．0-1 問題 P_0 では $u_j=1$，整数値問題 P では $u_j=[b/a_j]$ とする．$F_k(y)$ は，与えられた n 個の品物のうち，はじめの k 個だけを対象とし，ナップザックのサイズをパラメータ y とした問題の最適値である．任意の整数 $k(1 \leqq k \leqq n)$ と任意の $y(0 \leqq y \leqq b$(注3)$)$ に関して，$F_k(y)$ は再帰的に，

$$F_k(y) = \max_{\substack{y-a_k x_k \geqq 0 \\ x_k=0,\cdots,u_k}} \{c_k x_k + F_{k-1}(y-a_k x_k)\} \tag{3.2}$$

で与えられる[3]．初期条件は $F_0(y)=0$ である．ナップザック問題の最適値 z^* は $z^*=F_n(b)$ となる．$F_n(b)$ を求めるに要する計算量は，(3.2)の max 演算で数えると $O(b\sum_{j=1}^{n} u_j)$ である．演算には必要な記憶メモリーは $O(nb)$ である．

整数値ナップザック問題に対しては，さらに計算効率のすぐれたアルゴリズムが開発されている[11]．いま，

$$f(y) = \max_{x_1,\cdots,x_n} \left\{ \sum_{j=1}^{n} c_j x_j \,\middle|\, \sum_{j=1}^{n} a_j x_j \leqq y,\ x_j=0,\cdots,u_j \right\} \tag{3.3}$$

なるパラメトリックな問題を考える．明らかに，$f(y)=F_n(y)$ である．$f(y)$ は，パラメータ y の関数という意味でナップザック関数(knapsack function)とよばれ，任意の $y(0 \leqq y \leqq b)$ に対して，

(注1) 厳密には，$X_I=\{x \in R^n \mid Ax=b, x$ は非負整数$\}$ としたとき，$X_I \neq \phi$ かつ X_I は有界の条件が必要．

(注2) $\sum_{j=1}^{n} a_j \leqq b$ のときは，$\bar{x}_j=1(1 \leqq j \leqq n)$ となり，$\bar{P}_0$ の最適解が P_0 の最適解でもある．

(注3) $a_1, a_2, \cdots, a_n$ と b は整数と仮定し，$y=0,1,\cdots,b$.

$$f(y)=\max_{k\in Q(y)}[0,\ c_k+f(y-a_k)\ ;\ a_k\leqq y] \qquad (3.4)$$

と再帰的に表現できる．初期条件は $f(0)=0$ であり，$Q(y)=\{1,\cdots,n\}$ としてもよいが，さらに計算効率を改善する $Q(y)$ の定義が［9］で与えられている．再帰式(3.4)は(3.2)と比較して，DPの段階(stage)と状態(state)を入れ換えたものと考えられる．最適値 $z^*=f(b)$ を求めるのに要する max 演算の回数が $O(nb)$ であるばかりでなく，記憶メモリーも $O(b)$ で済み，(3.2)よりはるかにすぐれたアルゴリズムである．

DPによるアルゴリズムは，ナップザックのサイズ b が比較的小さいときは有効だが，b が大きくなると計算量が膨大になる．(2.1)の仮定のもとで整数値ナップザック問題 P に対応するナップザック関数 $f(y)$ について，

定理(Gilmore & Gomory[11])

ある $y_0\geqq 0$ が存在し，$y\geqq y_0$ なるすべての y 対して，関数 $\varphi(y)=\rho_1 y-f(y)$ は周期関数(周期 a_1)となる．すなわち，$\varphi(y)=\varphi(y-a_1)$ である．

y_0 の値に関して，$\rho_1>\rho_2$ ならば $y_0=\langle c_1/(\rho_1-\rho_2)\rangle$ とし，$\rho_1=\rho_2$ ならば $y_0=a_M(a_1-1)$ とすれば十分である[32]．ただし，$\langle x\rangle$ は x を下回らない最小の整数，$a_M=\max_{j\geqq 2}a_j$ である．定理が成立するできるだけ小さな y_0 の値を確定しようとする研究が続けられている[17]．

この定理によれば，$0\leqq y\leqq y_0$ なるすべての y に対して $f(y)$ の値を知れば，$y>y_0$ なる y に関する $f(y)$ の値については，$k=\langle(y-y_0)/a_1\rangle$ としたとき，

$$f(y)=f(y-ka_1)+kc_1 \qquad (3.5)$$

で求められる．k の定義より，$y-ka_1\leqq y_0$ は明らか．

分枝限定法

いま 0-1 ナップザック問題 P_0 を考える．分枝限定法は視覚的に「木構造」で表現され，各ノードに P_0 の限定された部分問題 SP_0 を対応させる．$N=\{1,\cdots,n\}$ とし，ノード m に対応して，

$N^1=\{j\in N\mid x_j=1$ に限定された変数$\}$

$N^0=\{j\in N\mid x_j=0$ に限定された変数$\}$

$F=\{j\in N\mid x_j=0$ または 1 の変数(自由変数)$\}$

なる互いに排反な部分集合を考えると，$x_j=1(j\in N^1)$ かつ $x_j=0(j\in N^0)$ の制約条件で限定された元問題 P_0 が，

部分問題 $SP_0(m)$:

$$v_0(m)=c_0+\max\Big\{\sum_{j\in F}c_jx_j\Big|\sum_{j\in F}a_jx_j\leqq b_0,\ x_j=0 \text{ or } 1,\ j\in F\Big\} \qquad (3.6)$$

となる．ここで，$c_0=\sum_{j\in N^1}c_j$，$b_0=b-\sum_{j\in N^1}a_j$ である．部分問題 SP_0 も 0-1 ナップザック問題であり，$F=N$(当然 $N^1=N^0=\phi$)と置いた部分問題 SP_0 が元問題 P_0 である．もし $b_0<0$ ならば SP_0 は実行不可能である．

第 1.2 節で述べたように，自由変数 $x_j(j\in F)$ を連続変数 $0\leqq x_j\leqq 1$ で置き換えた連続緩和問題 $\overline{SP_0}(m)$ を考えると，(2.4)と(2.6)を適用し，$SP_0(m)$ の最適値の上界値 $v_0^u(m)$ と下界値 $v_0^l(m)$ が求められる．

［分枝限定法によるアルゴリズム］

1. ノード $m=1$ に，$N^1=N^0=\phi$ と $F=N$ を対応させ，緩和問題 $\overline{SP_0}(1)$ を解き，ラベル $v_0^u(1)$ と $v_0^l(1)$ を付す．現在の最良値 $z_0=v_0^l(1)$，未分枝ノード集合 $\mathcal{N}=\{1\}$ として，2へ進む．
2. 任意のノード $m\in\mathcal{N}$ を選び，$\mathcal{N}=\mathcal{N}-\{m\}$ として3へ進む．もし $\mathcal{N}=\phi$ ならば5へ進む．
3. (a) $v_0^u(m)\leqq z_0$ ならば2へもどる．
   (b) $v_0^u(m)\leqq v_0^l(m)$ ならば，$z_0=v_0^l(m)$ として2へもどる．
   (c) $v_0^u(m)>v_0^l(m)$ ならば，任意の $k\in F$ を選び4へ進む．
4. (a) $m=m+1$，$\mathcal{N}=\mathcal{N}+\{m\}$，$N^0=N^0+\{k\}$，$F=F-\{k\}$ とする．緩和問題 $\overline{SP_0}(m)$ を解き，ノード m にラベル $v_0^u(m)$ と $v_0^l(m)$ を付す．4(b)へ進む．
   (b) $m=m+1$，$\mathcal{N}=\mathcal{N}+\{m\}$，$N^1=N^1+\{k\}$，$F=F-\{k\}$ とする．緩和問題 $\overline{SP_0}(m)$ を解き，ノード m にラベル $v_0^u(m)$ と $v_0^l(m)$ を付す．2へもどる．
5. 現在の z_0 が最適値 z_0^*，z_0 を与える解が最適解である．

分枝限定法で最も中心的役割を果たすのが，

1) 分枝ノード $m\in\mathcal{N}$ の選択(ステップ2)
2) 分枝変数 x_k，$k\in F$ の選択(ステップ3(c))

の二つのステップである．分枝ノード m の選択には，

N1) 最大の上界値 $v_0^u(m)$ をもつノード m(界値探索)
N2) 最後に生成されたノード m(線形探索)

の二つの規則がよく使われ，また分枝変数 x_k の選択には，

V1) $\rho_j=c_j/a_j(j\in F)$ の最大値に対応する変数
V2) 緩和問題 $\overline{SP_0}(m)$ を解いたときのピボット変数 $\bar{x}_p$

の二つの規則がよく使われている．

Kolesar[21]は，選択規則にN1とV1を用いたアルゴリズムを提案し，100変数の問題を解いた．Greenderg & Hegerich[13]は，N1とV2の組合せ(Greenbergの分枝限定法)およびN2とV2の組合せ(Greenbergの分枝探索法)の二つを提案し，20～50変数の数値例で，Kolesarの方法と比較実験した．その結果，生成されたノード数および計算時間の両方の点で，N2とV2を組

合せた分枝探索法が最もすぐれていたと報告している．Ahrens & Finke[1]は，N2とV1を組合せた場合を，Greenberg の分枝探索法と比較実験し，どちらの方法でも効率に大差ないと報告している．

1.4 最近の話題と傾向

ナップザック問題の分野における最近の話題の中で，主なもの五つを取り上げ簡単に説明しよう．

［A］ 問題サイズの縮小(Reduction)

アルゴリズムの面での大きな話題は，0-1 ナップザック問題の問題サイズ n の縮小に関するものである．いま問題 P_0 について，仮定(2.1)が成立し，その最適解 $x^*=(x_1^*,\cdots,x_n^*)$ が既知とする．このとき，

$$j_1=\min\{j\in N\mid x_j^*=0\}$$
$$j_2=\max\{j\in N\mid x_j^*=1\}$$
$$C=\{j_1, j_1+1,\cdots, j_2\}\subset N$$

とする．$x^*=(1,\cdots,1,0,\cdots,0)$ となる場合は，$j_1>j_2$ となり，$C=\phi$ と考える．Balas & Zemel[2]は，この添字集合 C を問題 P_0 の「コア」とよび，

コア問題 CP_0：

$$\max\Big\{\sum_{j\in C}c_jx_j\Big|\sum_{j\in C}a_jx_j\leqq b-\sum_{j<j_1}a_j,\quad x_j=0 \text{ or } 1,\ j\in C\Big\}\qquad(4.1)$$

を定義している．

元問題 P_0 の最適解 $x^*=(x_1^*,\cdots,x_n^*)$ とコア問題 CP_0 の最適解 $\hat{x}=(\hat{x}_1,\cdots,\hat{x}_n)$ との間には，$x_j^*=1\ (j<j_1)$，$x_j^*=\hat{x}_j\ (j\in C)$，$x_j^*=0\ (j>j_2)$ という関係が成立するという意味で，P_0 と CP_0 は等価である．われわれは，P_0 の代りに変数の数が少ない $CP_0(|C|\leqq n)$ を解けばよい．Balas & Zemel は，「係数 a_j と c_j をランダムに発生させた問題 P_0 について数値実験したところ，100変数の問題も 10,000 変数の問題も，コアのサイズ $|C|$ はいずれも25程度であった」という驚くべき事実を報告している．このことは，大規模な 0-1 ナップザック問題を解こうとする者を，大いに勇気づけるに違いない．

ところが，われわれは P_0 の最適解 x^* を知らないので，CP_0 を限定できない．そこで，$C\subset I$ という意味でコア C を近似する添字集合 I を求められないかという問題に到達する．Ingargiola & Korsh[16]や Nauss[29]などが，この問題に対する一つの解答を与えている．

定理(Nauss[29])

(2.1)の仮定の下で，$\bar{z}_0$ を問題 P_0 のある実行可能解に対応する目的関数値，$\bar{z}_0$ および $\bar{\lambda}$ をそれぞれ緩和問題 $\bar{P}_0$ の最適値および最適双対解 $(\bar{\lambda}=\rho_p)$ とするとき，

$$\beta_j\geqq\ \bar{z}_0-\hat{z}_0\ \longrightarrow\ x_j^*=1\qquad(4.2)$$
$$\beta_j\leqq-(\bar{z}_0-\hat{z}_0)\ \longrightarrow\ x_j^*=0\qquad(4.3)$$

を満たす問題 P_0 の最適解 $x^*=(x_1^*,\cdots,x_n^*)$ が存在する．ただし，$\beta_j=c_j-\bar{\lambda}a_j$ である．

この定理により，P_0 の一部の変数をあらかじめ0または1に固定(pegging)することができ，残りの変数の添字集合 $I_0=\{j\in N\mid|\beta_j|<\bar{z}_0-\hat{z}_0\}$ を含む最小の連続した添字集合を I とすれば，I がコア C の近似となっている．よって，われわれは $j\in I$(または I_0)なる 0-1 変数だけに縮小された問題(reduced problem)に対して，従来の分枝限定法やDPの手法を適用すればよい．

Nauss[29], Fayard & Plateau[7], Lauriere[25]などに計算結果が示されており，Lauriere は 60,000 変数の 0-1 ナップザック問題が IBM 370-168 で約30秒で解けたと報告している．

［B］ ハイブリッド型解法(Hybrid Algorithm)

ハイブリッド型解法は，Marsten & Morin[28]によって大系化されつつある分離可能な離散計画問題に対するアルゴリズムである．その基本思想は，DPの計算枠組に分枝限定法の限界値テストを組入れて，DPの状態空間を縮小し，計算効率の向上を計ろうとするものである．DPと分枝限定法を混合したという意味で，「ハイブリッド型解法」とよばれ，両者の長所を併せもつため，現在最も強力なアルゴリズムである．同様の研究に[22][37]がある．

［C］ 分割と分割数(Partition, Number of Partitions)

整数値ナップザック問題 P の実行可能解集合 X_I は，

$$X_I=\Big\{(x_1,\cdots,x_n)\in R^n\Big|\sum_{j=1}^n a_jx_j=b,\ x_j\text{ は非負整数}\Big\}\qquad(4.4)$$

と表わされるが，実行可能解 $(x_1,\cdots,x_n)\in X_I$ を $(a_1,\cdots,a_n)$ に限定された b の「分割」，またカージナル数 $|X_I|$ を「分割数」という．なお，不等号制約にはスラック変数を加え，等号制約に直して考える．

Horowitz & Sahni[15]は，すべての分割を求めるのに $\min(2^n,\ n|X_I|)$ の計算量を要するアルゴリズムを示している．Hayashi[14]は，他のいかなる分割の凸結合でも表わし得ない分割を極分割とよび，極分割だけを求めるアルゴリズムを提案している．問題 P の最適解は極分割であるから，後者のほうが無駄がない．

Lambe[24]は，分割数 $|X_I|$ が，

$$\binom{b+n}{n}\prod_{j=1}^n\frac{1}{a_j}\leqq|X_I|\leqq\binom{b+n\bar{a}}{n}\prod_{j=1}^n\frac{1}{a_j}\qquad(4.5)$$

であることを示した．ただし，$\bar{a}=\sum_{j=1}^n a_j/n$ である．

［D］ 貪欲解(Greedy Solution)の最適性

両替問題は整数値ナップザック問題の一種の拡張である．この両替問題に対して，Magazine et al.[27],

Tien & Hu[34]などが，任意の b に関して貪欲解が常に最適となるための，係数 a_j および $c_j(j=1,\cdots,n)$ の値の必要十分条件を与えている．さらに Tien & Hu は，与えられた係数が条件を満たさない場合について，貪欲解と最適解の目的関数値の最大誤差にも言及している．

［E］ 多項式オーダーの近似解法
(Polynomial Approximation Algorithm)

ナップザック問題およびその拡張された問題に対して，問題サイズ n と与えられた精度 $\varepsilon>0$ に関して多項式オーダーの計算時間と記憶容量で，近似解を求めるアルゴリズムの開発が行なわれている．前もって，最悪の場合に必要となる計算時間と記憶容量がわかるので，ユーザー側からは非常に高く評価されるであろう．

Lawler[26]の近似解法によれば，

0-1 ナップザック問題： 計算時間 $O(n\log\varepsilon^{-1}+\varepsilon^{-4})$
記憶容量 $O(n+\varepsilon^{-3})$

整数値ナップザック問題：計算時間 $O(n+\varepsilon^{-3})$
記憶容量 $O(n+\varepsilon^{-3})$

であるという．他に，［18］［31］［5］などの研究がある．

(すずき・ひさとし 東京工業大学)

参考文献

［1］ Ahrens, J. H. & G. Finke, "Merging and Sorting Applied to the Zero-One Knapsack Problem," *Opns. Res.* **23**(6), pp. 1099-1109(1975).

［2］ Balas, E. & E. Zemel, "Solving Large Zero-One Knapsack Problems," Management Science Research Report No. 408, Carnegie-Mellon Univ., 1976.

［3］ Bellman, R. E. & S. E. Dreyfus, Applied Dynamic Programming, Princeton Univ. Press, 1962.

［4］ Bradley, G. H., "Transformation of Integer Programs to Knapsack Problems," *Discrete Math.* **1**(1), pp. 29-45(1971).

［5］ Chandra, A. K., D. S. Hirschberg & C. K. Wong, "Approximate Algorithm for the Knapsack Problem and its Generalizations," IBM Research Report RC-5616, 1975.

［6］ Dantzig, G. B., "Discrete-Variable Extremum Problems," *Opns. Res.* **5**(2), pp. 266-277 (1957).

［7］ Fayard, D. & G. Plateau, "Resolution of the 0-1 Knapsack Problem: Comparison of Methods," *Math. Prog.* **8**, pp. 272-307(1975).

［8］ Feuerman, M. & H. Weiss, "A Mathematical Programming Model for Test Construction and Scoring," *Management Sci.* **19**(8), pp. 961-966(1973).

［9］ Garfinkel, R. S. & G. L. Nemhauser, Integer Programming, John Wiley & Sons, 1972.

［10］ Gilmore, P. C. & R. E. Gomory, "A Linear Programming Approach to the Cutting-Stock Problem —— Part II," *Opns. Res.* **11**(6), pp. 863-887(1963).

［11］ ——, & ——, "The Theory and Computation of Knapsack Functions," *Opns. Res.* **14**(6), pp. 1045-1074(1966).

［12］ Glover, F., "New Results on Equivalent Integer Programming Formulations," *Math. Prog.* **8**, pp. 84-90(1975).

［13］ Greenberg, H. & R. L. Hegerich, "A Branch Search Algorithm for the Knapsack Problem," *Management Sci.* **16**(5), pp. 327-332 (1970).

［14］ Hayashi, Y., "On Finding All Extreme Partitions," *Keio Eng. Reports* **31**(8), 1978.

［15］ Horowitz, E. & S. Sahni, "Computing Partitions with Applications to the Knapsack Problem," *JACM* **21**(2), pp. 277-292(1974).

［16］ Ingargiola, G. P. & J. F. Korsh, "Reduction Algorithm for Zero-One Single Knapsack Problems," *Management Sci.* **20**(4), pp. 460-463(1973).

［17］ Iwamura, K., "On Some Theorems of Knapsack Function," *JORSJ* **17** (4), pp. 173-183 (1974).

［18］ Johnson, D. S. "Approximation Algorithms for Combinatrial Problems," *J. Comput. System Sci.* **9**, pp. 256-278(1974).

［19］ Kendall, K. & S. Zionts, "Solving Integer Programming Problems by Aggregating Constraints," *Opns. Res.* **25**(2), pp. 346-351(1977).

［20］ Kianfar, F. "Stronger Inequalities for 0-1 Integer Programming: Computational Refinements," *Opns. Res.* **24**(3), pp. 581-585(1976).

［21］ Kolesar, P. J., "A Branch and Bound Algorithm for the Knapsack Problem," *Management Sci.* **13**(9), pp. 723-735(1967).

［22］ Kovacs, L. B., "Solution of Linear Integer Programming Problems by Dynamic Program-

ming," *Math. Operationsforsch. u. Statist.* **5** (3), pp. 163-176(1974).

[23] Kraft, D. H. & T. W. Hill, "The Journal Selection Problem in a University Library System," *Management Sci.* **19**(6), pp. 613-626 (1973).

[24] Lambe, T. A., "Bounds on the Number of Feasible Solutions to a Knapsack Problem," *SIAM J. Appl. Math.* **26**(2), pp. 302-305(1974).

[25] Lauriere, M., "An Algorithm for the 0/1 Knapsack Problem," *Math. Prog.* **14**, pp. 1-10 (1978).

[26] Lawler, E. L., "Fast Approximation Algorithms for Knapsack Problems," Memorandum No. UCB/ERL M77/45, Electronics Research Labo., Univ. of California, Berkeley.

[27] Magazine, M. J., G. L. Nemhauser & L. E. Trotter, Jr., "When the Greedy Solution Solves a Class of Knapsack Problems," *Opns. Res.* **23**(2), pp. 207-217(1975).

[28] Marsten, R. E. & T. L. Morin, "A Hybrid Approach to Discrete Mathematical Programming," *Math. Prog.* **14**, pp. 21-40(1978).

[29] Nauss, R. M., "An Efficient Algorithm for the 0-1 Knapsack Problem," *Management Sci.* **23**(1), pp. 27-31(1976).

[30] Pierce, J. F., "Application of Combinatrial Programming to a Class of All-Zero-One Integer Problems," *Management Sci.* **15**(3), pp. 191-209(1968).

[31] Sahni, S., "Approximate Algorithm for the 0/1 Knapsack Problem." *JACM* **22**(1), pp. 115-124(1975).

[32] Salkin, H. M., Integer Programming, Addison-Wesley, 1975.

[33] ——, & C. A. Dekluyver, "The Knapsack Problem:A Survey," *Nav. Res. Logist. Quart.* **22**(1), pp. 127-144(1975).

[34] Tien, B.N. & T. C. Hu, "Error Bounds and the Applicability of the Greedy Solution to the Coin-Changing Problem," *Opns. Res.* **25** (3), pp. 404-418(1977).

[35] Tillman, F. A. & J. M. Liittschwager, "Integer Programming Formulation of Constrained Reliability Problem," *Management Sci.* **13**(11), pp. 887-899(1967).

[36] Weingartner, H. M., Mathematical Programming and the Annalysis of Capital Budgeting Problems, Princeton-Hall, 1963.

[37] 鈴木久敏,「A Hybrid Approach to 0-1 Knapsack Problem」1977年6月, 整数計画法研究部会資料.

第2部　集合被覆問題および集合分割問題

2.1　集合被覆問題と集合分割問題の諸性質

集合 $I=\{1,\cdots,m\}$ の部分集合の一つのクラス $P=\{P_1,\cdots,P_n\}$, $P_j\subseteq I$, $j\in J=\{1,\cdots,n\}$ を考えよう. J の部分集合 $J^*\subseteq J$ に対し

$$(1)\qquad \bigcup_{j\in J^*} P_j=I$$

のとき J^* を P による集合 I の被覆(cover)と言う.

さらに,

$$(2)\quad j,\ k\in J^*,\ j\neq k \Rightarrow\ P_j\cap P_k=\phi$$

なる J^* を P による集合 I の分割(partition)と言う.

各 P_j に対して非負整数 c_j (コストとよぶ) が与えられたとき被覆 J^* の全コストを $\sum_{j\in J^*} c_j$ とする. (1)を満足する J^* のうちで全コスト最小な J^* を求める問題を**集合被覆問題**(SC)といい, (1)と(2)の両方を満足する J^* のうちで全コスト最小な J^* を求める問題を**集合分割問題**(SP)という. c_j がすべての j に対して一定値のとき**単一コスト問題**であるという.

0-1 定数 a_{ij} を,

$$a_{ij}=\begin{cases}1, & i\in P_j \text{ の時}\\ 0, & \text{そうでない時}\end{cases}$$

とし 0-1 変数 x_j を,

$$x_j=\begin{cases}1, & j\in J^* \text{ の時},\\ 0, & j\notin J^* \text{ の時},\end{cases}$$

とすれば, (SC)はつぎの 0-1 整数計画問題:

$$(3)\quad \text{最小化}\quad \sum_{j=1}^{n} c_j x_j$$

$$(4)\quad \text{条　件}\quad \sum_{j=1}^{n} a_{ij}x_j\geq 1,\quad i=1,\cdots,m$$

$$(5)\qquad x_j\in\{0,1\},\quad j=1,\cdots,n$$

を解くことになる. (SP)は同様にして式(4)を

$$(4')\qquad \sum_{j=1}^{n} a_{ij}x_j=1,\quad i=1,\cdots,m$$

に取替えればよい.

例 1：配送会社が毎日4カ所の無人販売機に商品を補充にゆく仕事を考えよう. 各無人販売機S1, S2, S3, S4に通じるルートは R1, R2, R3, R4, R5 の5通

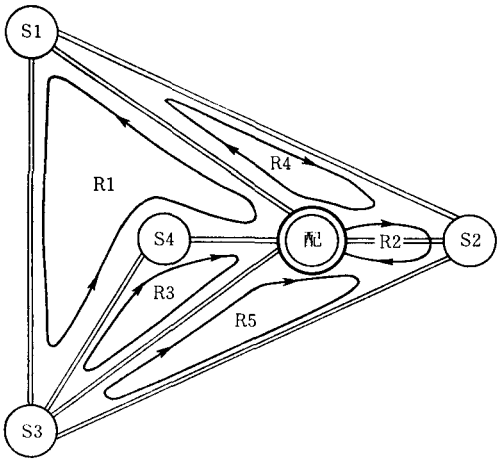

図 1 配送計画問題のルート(例1)

りあって，ルート R1 を使って S1，S3，S4 が補充でき費用は 2000 円であるものとする．また R2 を使って S2 が補充でき費用 3000 円，R3 は S3，S4 を費用 6000 円で，R4 は S1，S2 を費用 1000 円で，R5 は S2, S3 を費用 5000 円で補充できるものとする(図1参照)．このとき全費用が最小な配送計画は，$m=4$, $n=5$,

$$c=(2,\ 3,\ 6,\ 1,\ 5)$$

$$A=\begin{pmatrix}1&0&0&1&0\\0&1&0&1&1\\1&0&1&0&1\\1&0&1&0&0\end{pmatrix}$$

の (SC) を解けばよい．もし無人販売機を1回だけ訪れるような計画を作りたいならば同じ c, A をもつ (SP) を解けばよい．この他に(SC)または(SP)になる例として航空機要員計画[1]，バル・ルーティング[13]等がある．

(SP)はコストを変えるだけで(SC) を解くことに帰着できることが知られている[10, 21]：

定理 もし(SP)が解をもつならばコスト行 c' を $c_j'=c_j+L\sum_{i=1}^{m}a_{ij}$ とした (SC) の最適解の集合は対応する (SP)の最適解の集合と一致する．ここで L は $\sum_{j=1}^{n}c_j$ より大な自然数である．

また行列Aの行または列を除去して行列Aのサイズを小さくすることができる．詳細は文献[10, 12]を参照されたい．

被覆 J' の要素 j^* に対して $J'-\{j^*\}$ が再び被覆になるとき j^* は余分であるという．余分な j^* をもつ被覆は余分な被覆であるといい，余分でない被覆を**素被覆** (prime cover)という．定義から J' が素被覆であることは，

(16) $J'\ni\hat{j}$ などんな $\hat{j}$ に対しても，

$$\{i\mid\sum_{j\in J'}a_{ij}=1,\ i\in P_{\hat{j}}\}\neq\phi$$

と同値である．一般に被覆 J' から作られる $x_j'=1(j\in J')$，$x_j'=0(j\notin J')$ な x' を**被覆解**とよぶ．素被覆に対する x' を**素被覆解**とよぶ．素被覆の作り方については文献[4，10]を見られたい．

例 2

$$\begin{aligned}&x_1+x_2\quad\geq1\\&\quad x_2+x_3\geq1,\\&x_1\quad+x_3\geq1\\&x_1,x_2,x_3\in\{0,1\}\end{aligned}$$

において(1, 1, 1)は素でない被覆解であり，(0, 1, 1)，(1, 0, 1)，(1, 1, 0)は素被覆解である．

2.2 分枝限定法による集合被覆問題の解法

分枝限定法による解法には下界の計算戦略，分枝戦略等によりいくつものバリエーションがあるが，ここでは Lemke, Salkin, Spielberg の方法 [21, 10] を紹介する．ノード k における部分解を $W_k=\{j\mid x_j=0$ または1と決定された$\}$，

$S_k^+=\{j\mid j\in W_k$ かつ $x_j=1\}$，

$S_k^-=\{j\mid j\in W_k$ かつ $x_j=0\}$，

$F_k=\{j\mid j\notin W_k\}$，$Q_k=\{i\mid a_{ij}=0,\ \forall j\in S_k^+\}$，

$\bar{z}_0=$ いままでに得られた整数解のうちで最良なものの目的函数値，

で表わす．始めに $\bar{z}_0=\infty$，$W_0=\phi$ とする．ノード k での部分問題は，

$$(CP)_k\quad\text{最小化}\quad z_k=\sum_{j\in F_k}c_jx_j+\sum_{j\in S_k^+}c_j$$

$$\text{条　件：}\quad\sum_{j\in F_k}a_{ij}x_j\geq1(i\in Q_k)$$

$$x_j\in\{0,1\}(j\in F_k)$$

となる．$(CP)_k$ に対応するLP問題$(CP)'_k$ の最適解を x^*，最適値を Z_k^* とするとつぎのいずれかが成立する．

(i) $(CP)'_k$ の最適解 x^* は整数解ではない．

(ii) $(CP)'_k$ の最適解 x^* は整数解である．

(iii) $(CP)'_k$ は実行可能解をもたない．このケースが起るのは $\sum_{j\in F_k}a_{ij}=0$ な $i\in Q_k$ が存在するときである．このときは $z_k^*=\infty$ とおく．

(ii)，(iii)のときはバックトラックし $\bar{z}_0=\min\{\bar{z}_0,\ z_k^*\}$ とリセットする．(i)のときは，もし $\langle z_k^*\rangle\geq\bar{z}_0$ ならばバックトラックし，もし $\langle z_k^*\rangle<\bar{z}_0$ ならば $x_j'=\langle x_j^*\rangle$ $(j\in F_k)$によって作られる$(CP)_k$の被覆解 x' より $(CP)_k$ の素被覆解 x^k を求める．つぎに $\bar{z}_0=\min\{\bar{z}_0,\ z^k\}$ とリセットし $S_{k+1}^+=S_k^+\cup J^k$，$S_{k+1}^-=S_k^-$ とした部分解 W_{k+1} へ分枝する．ここで $z^k=\sum_{j\in J^k}c_j+\sum_{j\in S_k^+}c_j$，$J^k=\{j\mid x_j^k=1\}$ である．W_{k+1} は S_k^+，J^k に対応するから

分枝と同時に直ちにバックトラックする．

例 3

最小化　$Z_0=6x_1+8x_2+4x_3+3x_4+5x_5$

条件：
$$\begin{aligned} &x_1 + x_2 + x_4 \geq 1\\ &x_1 + x_3 + x_5 \geq 1\\ &x_2 + x_5 \geq 1\\ &x_1 + x_2 + x_3 \geq 1\\ &x_j=0 \text{ または } 1,\ j=1,\cdots,5 \end{aligned}$$

先ず $W_0=\phi$．$(CP)_0'$ の最適解は $x^*=(\frac{1}{2},\frac{1}{2},0,0,\frac{1}{2})$．$z_1^*=9.5<\bar{z}_0=\infty$．$x^*$ を切上げて被覆解 $x'=(1,1,0,0,1)$ を作る．これから素被覆解 $x^0=(1,0,0,0,1)$ が見つかる（(6)を満たすことに注意せよ）．$\bar{z}_0=\min\{\bar{z}_0, c\bar{x}_0\}=11$．$J^0=\{1,5\}$ だから $W_1=(1,5)$ に対応する部分解へ分枝する．直ちに $W_2=(1,\underline{5})$ へバックトラックする．ただし W の表現において i は $x_i=1$ を示し $\underline{i}$ は $x_i=0$ を示すものとする．

$(CP)_2'$　最小化　$Z_2=6+8x_2+4x_3+3x_4$

条件：
$$\begin{aligned} &x_2 \geq 1\\ &x_j\geq 0,\ j=2,3,4\text{(以後 } x\geq 0 \text{ と略記)} \end{aligned}$$

で $z_2^*=14$，$x^*=(1,1,0,0,0)$ 整数解．そこで再び $W_3=(\underline{1})$ へバックトラックする．$\bar{z}_0=\min\{11, z_2^*\}=11$．

$(CP)_3'$　最小化　$z_3=8x_2+4x_3+3x_4+5x_5$

条件：
$$\begin{aligned} &x_2 + x_4 \geq 1\\ &x_3 + x_5 \geq 1\\ &x_2 + x_5 \geq 1\\ &x_2 + x_3 \geq 1, x\geq 0 \end{aligned}$$

$(CP)_3'$ の最適解 $x^*=(\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2})$ で $z_3^*=10<\bar{z}_0$．そこで被覆 $(1,1,1,1)$ から素被覆 $x^3=(0,1,1,1)$ を作り $W_4=(\underline{1},3,4,5)$ へ分岐する．直ちに $W_5=(\underline{1},3,4,\underline{5})$ へ分枝する．$\bar{z}_0=\min\{11, cx^3\}=11$．

$(CP)_5'$　最小化　$z_5=7+8x_2$

条件：
$$\begin{aligned} &x_2\geq 1,\ x_2\geq 0 \end{aligned}$$

$z_5^*=15$ 整数解だから $W_6=(\underline{1},3,\underline{4})$ へ分枝．$\bar{z}_0=\min\{\bar{z}_0, z_5^*\}=11$．

$(CP)_6'$　最小化　$z_6=4+8x_2+5x_5$

条件：
$$\begin{aligned} &x_2 \geq 1\\ &x_2 + x_5\geq 1, \end{aligned} \quad x\geq 0$$

$z_6^*=12$ 整数解だから $W_7=(\underline{1},3)$ へ分枝．$\bar{z}_0=\min\{\bar{z}_0, z_6^*\}=11$．

$(CP)_7'$　最小化　$z_7=8x_2+3x_4+5x_5$

条件：
$$\begin{aligned} &x_2 + x_4 \geq 1\\ &x_5\geq 1\\ &x_2 + x_5\geq 1,\\ &x_2 \geq 1 \end{aligned} \quad x\geq 0$$

$z_7^*=13$ 整数解でバックトラックし $\bar{z}_0=\min\{\bar{z}_0, z_7^*\}=11$ で探索は終了する．最適被覆解は $x_1=x_5=1$，$x_j=0$，$j\neq 1$，5であり最適目的関数値は11である．

2.3　分枝限定法による集合分割問題の解法

文献[8, 9, 25, 26]によって始められ[15, 17, 18]によって追実験され，プログラム技術的に現在も改良が続けられている分枝限定法による一解法[9, 10]を紹介する．この方法は計算時間および記憶容量の双方から安定的な解法である．

与えられた問題を下のように階段形に変形する．各リスト内ではコストの小さい順にソートされるようにする．

$$\begin{aligned} \text{最小化}\quad & \overbrace{3x_1+7x_2}^{\text{リスト1}}+\overbrace{5x_3+8x_4+10x_5}^{\text{リスト2}}+\overbrace{4x_6+6x_7}^{\text{リスト3}}+\overbrace{9x_8}^{\text{リスト4}}\\ \text{条件：}\quad & x_1+x_2 = 1\\ & x_3+x_4+x_5 = 1\\ & x_5+x_6+x_7 = 1\\ & x_7+x_8 = 1\\ & x_2+x_4+x_6 = 1\\ & x_j\in\{0,1\},\ j=1,\cdots,8 \end{aligned}$$

I=行の添字の全体$=\{1,\cdots,m\}$とし部分解を S で表わす（W_k と書かない）．$S^+=\{j\,|\,x_j=1,\ j\in S\}$，$z(S)=\sum_{j\in S^+} c_j$ とする．S によって満足される制約式を $Q(S)$ とすれば $Q(S)=\sum_{j\in S^+} P_j$（$\sum$ は集合の直和を表わす）である．目下調べているカラム位置を j で示し，各リスト i のカラム位置を $\mathrm{ind}(i)$ で示す．m^*=リスト総数とする．

アルゴリズム

ステップ 1：$S=\phi$，$\bar{z}=\infty$

ステップ 2：$i^*=\min\{i\,|\,i\notin Q(S)\}$ とし，リスト i^* のトップ（すなわち左端）にインディケータを置き，この位置を j とする．すなわち j=リスト i^* の先頭位置，$\mathrm{ind}(i^*)=j$ とおく．

ステップ 3：j がリスト i^* を飛び出すかまたは $z(S)+c_j\geq\bar{z}$ のときはステップ 5 へゆく．そうでないときは $Q(S)\cap P_j=\phi$ を調べ非成立ならば $j=j+1$ かつ $\mathrm{ind}(i^*)=j$ とリセットしてステップ 3 の先頭へゆく．成立ならステップ 4 へゆく．

ステップ 4：$S^+=S^+\cup\{j\}$ とし $Q(S)=I$ を調べ非成立ならばステップ 2 へゆく．成立なら $\bar{z}=z(S)$ とリセットする．

ステップ 5：$S^+=\phi$ ならステップ 6 へゆく．そうでないなら $k=S^+$ の最後の要素，$S^+=S^+\backslash\{k\}$，$i^*=k$ が属していたリスト番号，$j=\mathrm{ind}(i^*)+1$，$\mathrm{ind}(i^*)=j$ とリセットしてステップ 3 へゆく．

ステップ 6：$\bar{z}=\infty$ なら分割は存在しない．そうでない時は最後に与えられた $\bar{z}$ に対応する分割が最適分割で

あり，zはその目的関数値である．停止．

Pierce[25] と Pierce & Lasky[26]はデータ構造を工夫しこの方法の効率改善を行なった．これは$m \leq 32$のデータに対して伊倉によって確認された[15]．岩村[17, 18] はこれらを総合してデータ域の記憶サイズが一般のm, n に対して約$90+2(n+1)<m/16>+20m+6n$バイトで与えられるプログラムを作成した．ただし固定小数点数は16ビットで持つものとし，$\langle y \rangle$はyを下回らない最小の整数を表わすものとする．

2.4 その他の解法について

Bellmore & Ratliff[4]は(SC)に対しその LP 基底のもつ特性を用いて条件制約式が1個ずつ増加するコードを作った．また Thiriez[29]は群論的アプローチによるコードを作った．Fulkerson, Nemhauser & Trotter[30] は$m=330$, $n=45$の(SC)問題がどうしても解けなかったことを報告している．ヒューリスティックなコードとしては Ignizio & Harnett[14], Roth[27]等がある．さらに Christofides & Korman[6]は単一コスト問題でもDPバウンドを使うことにより良い結果を得たと報告している．その他[1, 5, 7, 11, 13]等でもコード作成が行なわれているようである．

つぎに(SP)に対しては既述のもののほかに Salkin & Koncal[28]の結果が興味を引く．彼らは(SC)に直して dual all integer algorithm を適用したが，岩村[19]は(SC)に直さずに(SP)のままで dual all integer algorithm を適用することの有利性を考察した．Marsten[22]は Geoffrion らの指導の下に毎回 dual LP を下界値の計算に使う分枝限定法のコードを作成したが，分枝木の最下部でもLPを解くとか，バックトラッキングの不手際が目立つ．それにもかかわらず密度1.5%，$m=200$, $n=2,362$ の問題が解けたと報告している．Balas & Padberg[2, 3]は(SP)に対応する整数多面体の特性にもとづいた列生成法によるプライマルな解法を示したが，岩村&前田[16]の実験において追加すべきカラムが多過ぎて記憶域を使い過ぎる欠点が指摘されている．このほかに Michaud[23]の結果もある．

(SC)および(SP)は筆者の知る限りにおいて，ナップザック問題とともに整数計画問題の中では最も実用的コードが作成されてきた問題であると言える．とにかく，「このサイズの問題がこの時間内で解けた」という報告は整数計画の研究のはげみになってきたと思う．

ところで読者にも経験があることと思うが，しばしばプログラムに虫があることを知らずに実験していたり，プログラム化するときのプログラムデザイン・データ構造，プログラム技術の改良で効率を大幅に改善できたりすることがある．そこで筆者は(SC)，(SP)のアルゴリズムの研究に関しつぎの2点を主張したい．第1はソースプログラムの公開である．第2はプログラムデザイン書，プログラム設計書，フローチャートを研究者自らが作ることである．思うにソースプログラムが公開されるだけでも(SC)，(SP)の研究はもっと前進するであろう．

(いわむら・かくぞう　城西大学)

参考文献

[1] Arabeyre, J. P., J. Fearnley, F. C. Steiger & W.Teather, "The Airline Crew Scheduling Problem," *Transportation Science* **3**, pp. 140-163(1969).

[2] Balas, E. & M. W. Padberg, "On the Set-Covering Problem," *Opns. Res.* **20**, pp. 1152-1161(1972).

[3] Balas, E. & M. W. Padberg, "On the Set-Covering Problem: II. An Algorithm for Set Partitioning," *Opns. Res.* **23**, pp. 74-90(1975).

[4] Bellmore, M. & H. D. Ratliff, "Set Covering and Involutory Bases," *Man. Sci.* **18**, pp. 194-206(1971).

[5] Booler, J. M. P., "A Method for Solving Crew Scheduling Problems," *Opl. Res. Q.*, **26** (I), pp. 55-62(1975).

[6] Christofides, N. & S. Korman, "A Computational Survey of Methods for the Set Covering Problem," *Man. Sci.* **21**, pp. 591-599 (1975).

[7] Etcheberry, J., "The Set-Covering Problem: A New Implicit Enumeration Algorithm," *Opns. Res.* **25**, pp. 760-772(1977).

[8] Garfinkel, R. S. & G. L. Nemhauser, "Optimal Political Districting by Implicit Enumeration Techniques," *Man. Sci.* **16**, pp. B495-B508(1970).

[9] ——, & ——, "The Set-Partitioning Problem: Set Covering with Equality Constraints," *Opns. Res.* **17**, pp. 848-856(1969).

[10] ——, & ——, Integer Programming, John Wiley & Sons, 1972.

[11] Gondran, M. et J. L. Laurière, "Un Algorithme pour le Probleme de Partionnement," *R. A. I. R. O.*, **8**. V-1, pp. 27-40(1974).

[12] Guha, D. K., "The Set-Covering Problem with Equality Constraints," *Opns. Res.* **21**,

pp. 348-351(1973).

[13] Heurgon, E., "Un Probleme de Recouvrement: l 'Habillage des Horaires d'une ligne d'autobus," R. A. I. R. O. 6e annèe, V-1, pp. 13-29(1972).

[14] Ignizio, J. P., & R. M. Harnett, "Heuristically Aided Set-Covering Algotithms," *International J. of Computer and Information Sciences* **3**, pp. 59-70(1974).

[15] 伊倉義郎「Set Partitioning の Algorithm について」1975. 11. 4.

[16] Iwamura, K. & Y. Maeda, "A Computational Experiment of the Set Partitioning Algorithm proposed by Balas and Padberg," 1977, Josai Univ., Saitama.

[17] 岩村覚三「GN(内部仕様書) Version 1-Modification 2」1977年7月，整数計画法研究部会資料.

[18] 岩村覚三「集合分割問題に対する分岐限定法アルゴリズムの詳細な分析，第2版」1977年9月，整数計画法研究部会資料.

[19] 岩村覚三「集合分割問題を解く dual all integer algorithm に対する分析，第2版」1979年1月，整数計画法研究部会資料.

[20] Lawler, E. L., "Covering Problems: Duality Relations and a New Method of Solution," *SIAM J. Appl. Math.* **14**, pp. 1115-1132(1966).

[21] Lemke, C. E., H. M. Salkin & K. Spielberg, "Set Covering by Single-Branch Enumeration with Linear-Programming Sub-Problems," *Opns. Res.* **19**, pp. 998-1022(1971).

[22] Marsten, R. E., "An Algorithm for Large Set Partitioning Problems," *Man. Sci.* **20**, pp. 774-787(1974).

[23] Michaud, P., "Exact Implicit Enumeration Method for Solving the Set Partitioning Problem," *IBM. J. of R&D* **16**, pp. 573-578(1972).

[24] Nemhauser, G. L., L. E. Trotter, Jr. & R. M. Nauss, "Set Partitioning and Chain Decomposition," *Man. Sci.* **20**, pp. 1413-1421(1974).

[25] Pierce, J. F., "Application of Combinatorial Programming to a Class of All-Zero-One Integer Programming Problems," *Man. Sci.* **15**, pp. 191-209(1968).

[26] Pierce, J. F. & J. S. Lasky, "Improved Combinatorial Programming Algorithms for a Class of All-Zero-One Integer Programming Problems," *Man. Sci.* **19**, pp. 528-543(1973).

[27] Roth, R., "Computer Solutions to Minimum-Cover Problems," *Opns. Res.* **17**, pp. 455-465(1969).

[28] Salkin, H. M. & R. D. Koncal, "Set Covering by an All Integer Algorithm: Computational Experience," *JACM* **20**, pp. 189-193 (1973).

[29] Thiriez, "The Set Covering Problem: A Group Theoretic Approach," *R. I. R. O.* 5e *année*, V-3, pp. 83-104(1971).

[30] Fulkerson, D. R., G. L. Nemhauser & L. E. Trotter, Jr., "Two Computationally Difficult Set Covering Problems that Arise in Computing the 1-Width of Incidence Matrices of Steiner Triple Systems," *Math. Prog. Study* **2**, pp. 72-81(1974).

外国雑誌リスト (交換・寄贈により学会事務局にきているものです．ご利用ください．)

Aplikace Matematiky (Czechoslovakia)
Arkiv for Matematik (Sweden)
Cahiers du Centre Détudes de Recherche Operationnelle (Belgique)
Carnegie-Mellon University Research Report (U. S. A.)
Czechoslovak Mathematical Journal (Czechoslovakia)
Dissertations Mathematicae (Poland)
Ekonomicko-Matematicky Obzor (Czechoslovakia)
European Journal of Operational Research (The Netherlands)
FOA Report (Sweden)
Harvard Business Review (U. S. A.)
Hongkong Productivity News (Hong Kong)
Indagations Mathematicae (The Netherlands)
Interfaces, A TIMS-ORSA Journal (U. S. A.)
International Journal of Production Research (England)